IN THE UNITED STATES
PATENT AND TRADEMARK OFFICE

Patent Application

Inventors: Eric T. Bax

Case No.: iSpheres 7

Serial No.: 10/775,745 Group Art Unit: 2164

Filing Date: February 9, 2004

Examiner: Sathyanaraya R. Pannala

Title: Finite-State Machine Augmented For Multiple Evaluations of Text

MS Appeal Brief - Patents
Commissioner For Patents
PO Box 1450
Alexandria, VA 22313-1450

APPLICANT'S/APPELLANT'S APPEAL BRIEF

SIR:

Applicant/Appellant hereby appeals to the Board of Patent Appeals and Interferences in response to the Notice of Panel Decision from Pre-Appeal Brief Review mailed on December 23, 2008. The fee set forth in 37 CFR §41.20(b)(1) has been previously submitted in connection with the Request for Pre-Appeal Brief Request for Review. The fees set forth in 37 CFR §41.20 (b)(2) and a one-month Extension of Time to File to Deposit will be charged to PTO Deposit Account No. 501602.

A single copy of this Brief is being submitted pursuant to MPEP §1205.02.

TABLE OF CONTENTS

REAL PARTY IN INTEREST

The real party in interest is Avaya Inc., the assignee of the above-identified application, as evidenced by the assignment recorded in the US Patent and Trademark Office on Reel 018500, Frame 0269.

RELATED APPEALS AND INTERFERENCES

None.

STATUS OF CLAIMS

Claims pending: 1-6

Claims allowed: 6

Claims rejected: 1-3, 5

Claims objected to: 4

Claims appealed: 1-3, 5

STATUS OF AMENDMENTS

No amendments were filed subsequently to the notice of final rejection. A Response to Final Office Action that was filed on 22 January 2008 and that contains only Remarks/Arguments was entered. A Pre-Appeal Brief Request for Review that was filed on 11 March 2008 was entered. A Notice of Panel Decision from Pre-Appeal Brief Review was mailed on 23 December 2008, directing the Applicant to proceed to the Board of Patent Appeals and Interferences.

SUMMARY OF CLAIMED SUBJECT MATTER

Traditional techniques for determining how strongly a text matches a pattern (a set of character sequences representing a topic) use a counter. The counter generates a score that indicates how strongly the text matches the pattern. Computing a score for a text is called a counter evaluation. A counter may be evaluated by using a finite-state machine that represents the counter. A finite-state machine has states and transitions between the states. Each transition between states is represented by a pair of the form (character, next state). Each state has a value. To perform a counter evaluation, the finite-state machine accumulates in a sum the value of each state the machine visits.

Some finite state machines use multi-counters to evaluate text. A multi-counter contains a set of counters each corresponding to a different topic (i. e., a different pattern). As each pattern in the text is encountered that corresponds to a topic, the counter for that topic is updated (specification at page 3, lines 6-26). One method of multi-counter evaluation is to perform multi-counter evaluation sequentially for the counters in the multi-counter. However, this method is relatively slow (specification at page 4, lines 1-3). Another method is to use multiple computation resources to perform evaluations of two or more counters in the multi-counter at once. However, this method is relatively resource-intensive (specification at page 4, lines 1-4).

The invention is directed to performing multi-counter evaluations where multiple counters correspond to a single (merged) finite state machine that is augmented with state value lists instead of state values. Each state value list indicates which counters' scores receive which values for the state. This produces a list of counter scores. This allows multiple counters to be evaluated simultaneously using a single computational resource. This allows for faster and less resource intensive us of computational resources (specification at page 4, lines 7-14, 20-21).

Independent Claim 1

Independent claim 1 is directed to a method (figures 2 and 3) for performing multi-counter evaluation of a text (specification at page 4, lines 7). Claim 1 recites applying to the text a merged finite-state machine representing a plurality of single-counter finite-state machines, each representing a different one of a plurality of counters (specification at page 4, lines 9-11 and 20-21, , page 6, lines 16-18, figure 1), and wherein at least one state of the merged finite-state machine each corresponds to a multiplicity of states each of a different one of said single-counter finite-state machines (specification at page 6, line 21 (state 0 in second table), augmented with state value lists where each state value list indicates which counter of the multi-counter receives which value for the state of the merged finite-state machine (specification at page 6, line 25 (state 4 in second table), page 7, lines 3 and 5 (states 9 and 11 in table)); accumulating the values of the states of the merged finite-state machine separately for each counter of the multi-counter (specification at page 4, line 11, page 7, lines 21-22), thereby producing a list of counter scores (specification at page 4, lines 11-12, page 7, lines 21-22, figure 1, state, 9); and updating each counter with its counter score (specification at page 4, lines 1-13, page 7, lines 21-23).

Independent Claim 2

Independent claim 2 is directed to a method (figures 2 and 3) for performing multi-counter evaluation of a text (specification at page 4, lines 7-11). Claim 2 recites applying to the text a merged finite-state machine representing a plurality of single-counter finite-state machines, each representing a different one of a plurality of counters (specification at page 4, lines 9-11 and 20-21, page 6, lines 16-18, figure 1), and wherein at least one state of the merged finite-state machine each corresponds to a

multiplicity of states each of a different one of said single-counter finite-state machines (specification at page 6, line 21 (state 0 in second table)), augmented with state value lists where each state value list indicates which patterns in which counters of the multi-counter are found when the state of the merged finite-state machine is entered, producing a list of patterns for each counter (specification at page 13, lines 20-23); and updating each counter with its list of patterns (specification at page 13, lines 23-26).

Independent Claim 3

Independent claim 3 and dependent claim 4 are directed to a method (figure 2) for constructing a multi-counter finite-state machine augmented with state value lists (specification at page 6, lines 16-18, page 7, line 27-page 8, line 15). Claim 3 recites providing by computer an empty augmented finite-state machine that has only a start state, with no transitions and no value list (specification at page 7, line 27- page 8, line 2); accumulating by computer a finite-state machine of each counter of the multi-counter that corresponds to one or more pattern-amount pairs into the augmented finite-state machine to form a merged machine representing a plurality of single-counter finite-state machines each representing a different one of a plurality of counters (specification at page 3, line 19, page 7, line 27, page 8, lines 6-15, figure 1, state 0), and wherein at least one state of the merged finite-state machine each corresponds to a multiplicity of states each of a different one of said single-counter finite-state machines (specification at page 6, line 21 (state 0 in the second table), page 8, lines 6-10), including converting state values of states of the finite-state machines of the counters of the multi-counter into state-value lists of states of the merged machine , and updating the merged machine with the state-value lists (specification at page 8, line 17-page 9, line 26).

Independent Claim 5

Independent claim 5 is directed to a method for adding a pattern that consists of a single sequence of characters and a corresponding pattern value, from a counter to an augmented finite-state machine (specification at page 11, lines 8-11). Claim 5 recites, providing the pattern (Id.); providing the corresponding pattern value (specification at page 11, lines 13-16); providing the augmented finite-state machine having a plurality of machine states and representing a plurality of single-counter finite-state machines each representing a different one of a plurality of counters (specification at page 3, line 19, page 7, line 27, page 8, lines 6-15, figure 1, state 0), and wherein at least one state of the augmented finite-state machine each corresponds to a multiplicity of states each of a different one of said single-counter finite-state machines (specification at page 6, line 21 (state 0 in the second table), page 8, lines 6-10); advancing through the machine states by applying the machine to the sequence of characters as a text (specification at page 12, line 1); if the machine would halt when applied to the sequence of characters as a text, then adding states and transitions to the machine to prevent halting; forbearing from the adding if the machine would not halt when applied to the sequence of characters as a text (specification at page 12, lines 3-4); for a final state that would be reached by the machine supplemented with the added states and transitions (specification at page 12, lines 6-8), forming a state value list if the final state lacks a state value list, forbearing from forming a state value list if the final state has a state value list, and adding to the state value list a reference to the counter and the pattern (specification at page 12, lines 6-8); and updating the final state of the machine with the state value list (Id.).

GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

Rejection of claims 1-3 and 5 under 35 U.S.C. §103(a) over U.S. patent number 7,072,880 (Beasley) in view of U. S. patent number 5,630,130 (Perotto et al.).

ARGUMENTS

The disclosure of Beasley.

Beasley discloses a finite-state network. The finite-state network comprises a series of domains (Beasley, elements 350-354 in figures 8 and 9). Each domain 350-354 is an individual finite state machine. Each domain 350-354 has a counter which counts the number of states indicating the number of paths leading from the previous state (Beasley, col. 9, lines 32-39, and figure 9). The counts for each domain are separate and may be turned on or off (Beasley, col. 4, lines 60-65, and claim 16). A series of concatenated domains 350-354 are strung together for form the finite-state network (Beasley, col. 4, lines 43-46).

The finite-state network is used to parse through an input string. The input string contains a series of sub-strings/words. The sub-string/word start/end positions mark the boundaries of the domains (Beasley, col. 4, lines 35-43). The first domain 350 parses the input string and upon detecting a sub-string/word, returns an index that corresponds to the sub-string/word (Beasley, col. 9, lines 32-39, col. 10, lines 8-14). The index is a word-number mapping that is always the same and unique to the sub-string/word detected in the domain 350-354. Thus, each time a domain 350-354 detects a unique sub-string/word, the same index is returned. This process is repeated as the finite-state network transitions through each domain 350-354 (Beasley, col. 9, lines 32-56).

The disclosure of Perotto et al. (Perotto).

Perotto discloses a multitasking controller that has a separate program counter, a separate accumulator, and a separate index register, for each one of a plurality of tasks. Thus, there is a bank of a plurality of individual program counters, another bank of a plurality of individual accumulators, and another bank of a plurality of individual index registers.

A program counter points to a memory location containing the next instruction that is to be executed of its corresponding task (Perotto, col. 4, lines 4-6). An accumulator holds data associated with its corresponding task for use by the arithmetic and logic unit (Perotto, col. 5, lines 31-34). An index register stores an address employed by its corresponding task as an offset value for indexed addressing (Perotto, col. 5, lines 1-2 and 38-40).

The rejection of claims 1-3, and 5

The claimed invention is directed to multi-counter evaluation. According to the claims, multi-counter evaluation is performed using a merged finite-state machine that represents the finite-state machines of the individual counters of the multi-counter. The merged finite-state machine is augmented with state value lists instead of state values. Each state value list indicates which counter scores receive which values for the corresponding state.

In other words, claims 1-3 and 5 requires two separate finite-state machines sharing a common state in the merged machine. Looking at the specification (page 6, second table, and figure 1), figure 1 shows states 0-4 forming a first separate finite-state machine. States 0 and states 5-11 in figure 1 form a second finite state machine. Both the first and second state machines each have a separate counter (specification at page 6, lines16-27, and page 7, lines 1-7). The two merged finite-state machines also share a common state (state 0) that is the beginning state of both finite-state machines. The merged finite-state machine has a plurality counters (one for the first finite-state machine and one for the second finite-state machine) and share a common state (state 0) as required by claims 1-3 and 5.

The Examiner pointed to figures 13 and 14, col. 13, lines 56-67, and col. 14, lines 16-24 and 30-33 of Beasley as teaching a merged or augmented finite-state machine having at least one state each

corresponding to a multiplicity of states each of a different one of the single-counter finite state machines and augmented with state value lists. The Applicant respectfully disagrees.

Instead, Beasley teaches a finite-state network that contains a series of domains (finite-state machines) 350-354 (Beasley, figures 8 and 9). Each domain 350-354 has a counter that counts the number states indicating the number of paths leading from the previous state (Beasley, col. 9, lines 32-39). The counts/counters for each domain are separate and unique to each domain (Beasley, col. 4, lines 60-65). A single domain does not have a plurality of counters associated with the domain as required by claims 1-3 and 5. The domains are a series of state machines that are strung together serially (see, figures 8 and 9). The finite state-network corresponds to strings/words, while each state machine corresponds to a domain of sub-strings/formants. Since the finite-state network is a concatenation of finite-state machines, the finite-state network does not have at least one state each corresponding to a multiplicity of states each of a different one of the finite-state machines as required in claims 1-3 and 5.

Compare this to figure 1 from the application where each of the merged finite state machines has a common state (state 0). It could be argued that domain 350 in Beasley figures 8 and 9 shares a common state (the state represented by the number 4 in domain 350 in figure 9) and therefore reads on claims 1-3 and 5. However, domain 350 only has a single counter (Beasley, col. 9, lines 32-39, figure 9), not a plurality of counters as required in claims 1-3 and 5. Beasley sets forth steps for the preparation and use of a network for substring-to-number mapping and for retrieving related information, and not what the Examiner purports.

In order for Beasley to disclose what is in claims 1-3 and 5, two of the domains 350-354 would have to be merged. For example, if the node designated by 4 in domain 350 of Beasley figure 9 was merged with either one of the nodes designated with a 2 in domain 351 of Beasley figure 9 to

form a single common state as shown for state 0 in Figure 1 of the application, then Beasley would disclose what is claimed in claims 1-3 and 5. However, this is not the case.

Second, Beasley and Perotto fail to teach state value lists where each state value list indicates which **counter of the multi-counter receives which value** for the state of the merged finite-state machine as required by claim 1. The Examiner asserted that Beasley discloses state value lists in col. 13, lines 56-67 and col. 14, lines16-24. The Applicant respectfully disagrees.

Beasley discloses that a unique index or an array of labeled indices is returned when one or more substrings are identified, in col. 13, lines 56-67, and col. 14, lines16-24. The returned index/indices do not indicate which counter will receive which value. In fact, the returned index/indices have nothing to do with a counter, but instead are an index into a database (col. 14, lines 3-6).

Beasley does disclose that each domain has a separate counter that counts number states indicating the number of paths leading from the previous state (Beasley, col. 9, lines 32-39, figure 9). Since each domain has a separate counter, there is no indication of which counter will receive which value as required in claim 1. Beasley has no reason to indicate which counter to update because each domain only has a single counter and counter information is not shared/updated between domain counters.

Third, Beasley and Perotto fail to teach multi-counters as required by claims 1-3 and 5. The Examiner did concede that "Beasley does not explicitly teach using multi-counters," and cited Perotto for this teaching. But the combination of Beasley and Perotto likewise fails to teach Applicant's claimed invention.

As is evident from the description of Perotto given above, Perotto bears no relation either to Beasley or to Applicant's claimed invention. Perotto is concerned with efficient operation of a multitasking controller, and not with the substring-number mapping of Beasley or with multi-

counter evaluation as disclosed in the specification. Nor does Perotto disclose a multi-counter, as is made evident by the above discussion of the disclosure of Perotto. Nevertheless, even if one assumes for purposes of argument that Perotto does disclose a multi-counter, Perotto. still fails to cure the fundamental failure of Beasley to disclose, teach, or suggest the claimed invention. Specifically, the combined teachings of Beasley and Perotto fail to disclose, teach, or suggest a merged or augmented finite-state machine as defined in the claims, or at least a state-value list and augmentation therewith of the merged or augmented finite-state machine, as is required by the recitations of all of Applicant's claims.

Therefore, the combined teachings of Beasley and Perotto fail to disclose, teach, or suggest merged finite-state machine representing a plurality of single-counter finite-state machines sharing a common state, a state value list, or a multi-counter as defined in the claims.

Fourth, the Examiner summarily dismissed Applicant's explanation above by saying that "Applicant's arguments fail to comply with 37 CFR 1.11(b) because they amount to a general allegation that the claims define a patentable invention without specifically pointing out how the language of the claims patentably distinguishes them from the references."

This cursory dismissal is, at best, disingenuous. Applicant has explained at length why the combined teachings of the applied references "fail to disclose, teach, or suggest a merged or augmented finite-state machine as defined in the claims, or at least a state-value list and augmentation therewith of the merged or augmented finite-state machine, as is required by the recitations of all of applicant's claims," and has pointed out and quoted the specific recitations of the claims that distinguish his invention from the applied references. How much more specific could Applicant be?

Fifth, other than nakedly asserting that certain passages of Beasley and Perotto teach the claimed matter, the Examiner has provided no

explanation for his assertion of obviousness. Nor has the Examiner rebutted Applicant's contrary position. This is insufficient to establish obviousness. "[R]ejections on obviousness grounds cannot be sustained by mere conclusory statements; instead, there must be some articulated reasoning with some rational underpinning to support the legal conclusion of obviousness." KSR Int'l. v. Teleflex Inc., 550 U.S. (2007), citing In re Kahn, 441 F. 3d 977, 988 (Fed. Cir. 2006). Hence, the Examiner has failed to establish obviousness of Applicant's claimed subject matter.

CONCLUSION

For all of the reasons given above, the Applicant respectfully asserts that the Section 103(a) rejection of their appealed claims over Beasley in view of Perotto is not well founded. Applicant therefore respectfully requests that the rejection of the appealed claims be reversed.

Respectfully submitted,

Eric T. Bax

By _____

Douglas M. Grover
Corporate Counsel
Reg. No. 52,974
303-538-4926

Date: 2-4-2009

Avaya Inc.
Docket Administrator
1300 West 120th Ave
Room B1-F53
Westminster, CO 80234

Claims Appendix

The claims on appeal:

1. A method for performing multi-counter evaluation of a text, said method comprising computer-implemented steps of:

applying to the text a merged finite-state machine representing a plurality of single-counter finite-state machines each representing a different one of a plurality of counters and wherein at least one state of the merged finite-state machine each corresponds to a multiplicity of states each of a different one of said single-counter finite-state machines, augmented with state value lists where each state value list indicates which counter of the multi-counter receives which value for the state of the merged finite-state machine;

accumulating the values of the states of the merged finite-state machine separately for each counter of the multi-counter, thereby producing a list of counter scores; and

updating each counter with its counter score.

2. A method for performing multi-counter evaluation of a text, said method comprising computer-implemented steps of:

applying to the text a merged finite-state machine representing a plurality of single-counter finite-state machines each representing a different one of a plurality of counters and wherein at least one state of the merged finite-state machine each corresponds to a multiplicity of states each of a different one of said single-counter finite-state machines, augmented with state value lists where each state value list indicates which patterns in which counters of the multi-counter are found when the state of the merged finite-state machine is entered;

producing a list of patterns for each counter; and

updating each counter with its list of patterns.

3. A method for constructing a multi-counter finite-state machine augmented with state value lists, said method comprising the computer-implemented steps of:

providing by computer an empty augmented finite-state machine that has only a start state, with no transitions and no value list;

accumulating by computer a finite-state machine of each counter of the multi-counter that corresponds to one or more pattern-amount pairs into the augmented finite-state machine to form a merged machine representing a plurality of single-counter finite-state machines each representing a different one of a plurality of counters and wherein at least one state of the merged finite-state machine each corresponds to a multiplicity of states each of a different one of said single-counter finite-state machines, including

converting state values of states of the finite-state machines of the counters of the multi-counter into state-value lists of states of the merged machine, and

updating the merged machine with the state-value lists.

5. A method for adding a pattern that consists of a single sequence of characters and a corresponding pattern value, from a counter to an augmented finite-state machine, said method comprising computer-implemented steps of:

providing the pattern;

providing the corresponding pattern value;

providing the augmented finite-state machine having a plurality of machine states and representing a plurality of single-counter finite-state machines each representing a different one of a plurality of counters and wherein at least one state of the augmented finite-state machine each corresponds to a multiplicity of states each of a different one of said single-counter finite-state machines;

advancing through the machine states by applying the machine to the sequence of characters as a text;

if the machine would halt when applied to the sequence of characters as a text, then adding states and transitions to the machine to prevent halting;

forbearing from the adding if the machine would not halt when applied to the sequence of characters as a text;

for a final state that would be reached by the machine supplemented with the added states and transitions, forming a state value list if the final state lacks a state value list, forbearing from forming a state value list if the final state has a state value list, and adding to the state value list a reference to the counter and the pattern; and

updating the final state of the machine with the state value list.

Evidence Appendix

None

Related Proceedings Appendix

None